ECE 440 Final

Linxiao Bai

A

For a customer i, the time he finishes his call follows a exponential distribution: T_{ci} ~ exp(mu).

We want to prove that Tk as the additive result of all customer also follows an exponential distribution:

By problem setting TK=min(Tc1, Tc2, …, Tci).

That is: P(Tk<s)=1-$\prod_{for\ all\ i} \mathrm{P}(\mathrm{Tci} > \mathrm{s})$ because each customer is iid.

So, P(Tk>s)=1-exp(-k*mu*s)

By exponential cdf, this is a solid exponential distribution with parameter k*mu.

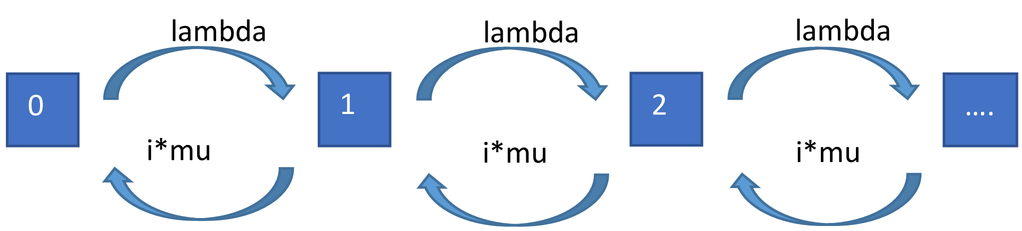
B

1. Because calling time is an exponential distribution, conditioning on time will not provide information. So the probability is 1/k as any line can finish first with equal probability.
2. Similarly, any of the two customer can hung up with equal probability, p= ½
3. By cdf of exp(3), p=1-(1-exp(1/3*3))≈0.37
4. Again conditioning on time provides no information, p=0.37

C

The problem can be modeled as an CTMC because the number of calling at time t depends only on the time before. Also because each calling time, and income call follows an exponential distribution with mu, the transition rate is time-invariant.

The process can be modeled as:



D

To model the transition probability, we apply normalizing factor (lambda +i*mu) to all:

Pij=

- lambda/(lambda +i*mu) , j=i+1
- i*mu/(lambda +i*mu) , j=i-1
- 0 , otherwise

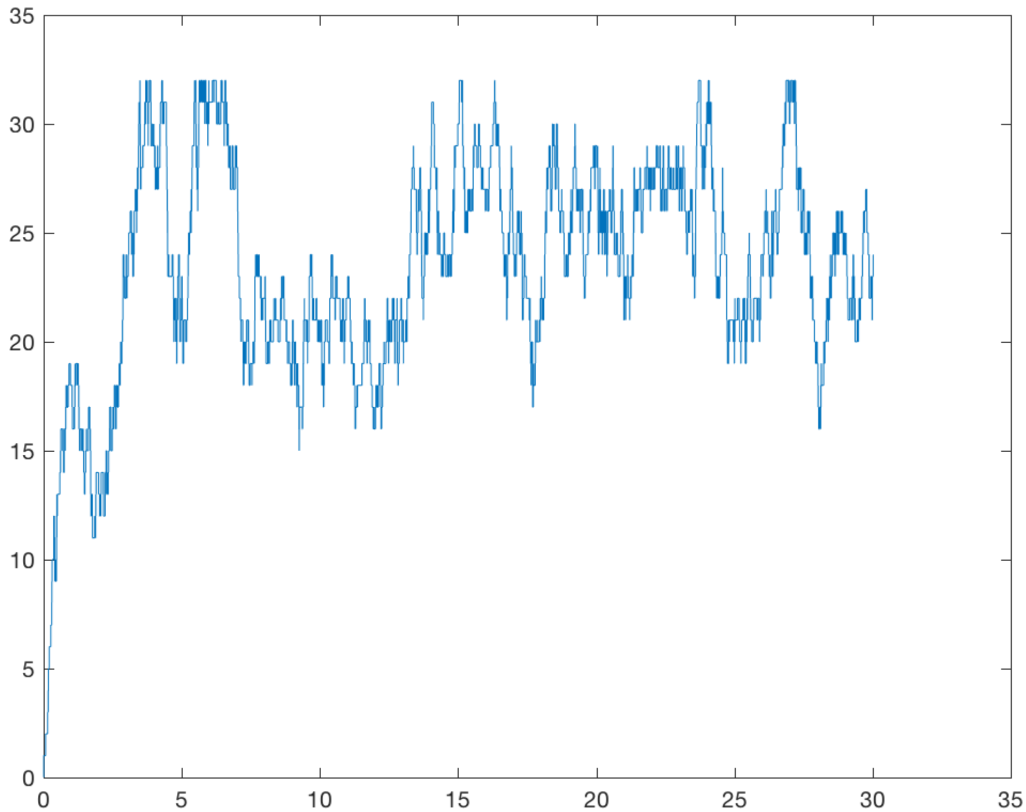
E

The model we build in D describe the transition probability of transition in uniform time interval t.

Also the CTMC is ergotic because 1. The model is formulate in a real-life problem. 2. It is irreducible, fully connected, and positive recurrent.

F

The plot shown below shows the simulation in half an hour. Where x axis is time in minute unit and y axis is number of call in that minute.



G

G: balanced equation:

$$\begin{cases} \lambda P_0 = \mu P_1 \\ \lambda P_1 = 2\mu P_2 \\ \vdots \\ \lambda P_{k-1} = k\mu P_k \end{cases} \Rightarrow \begin{array}{l} P_1 = (\lambda/\mu) P_0 \\ P_2 = (\lambda/\mu)^2 (1/2) P_0 \\ \\ P_k = (\lambda/\mu)^k (1/k!) P_0 \end{array}$$

apply: $\sum P_k = 1$:

$$\therefore 1 = \sum_{k=0}^{K} \left(\frac{\lambda}{\mu}\right)^k \frac{1}{k!} \cdot P_0$$

$$\Rightarrow P_0 = \left[\frac{1}{\sum^{K} \left(\frac{\lambda}{\mu}\right)^k \frac{1}{k!}}\right]$$

$$\therefore P_k = (\lambda/\mu)^k \cdot \frac{1}{k!} \cdot P_0$$

$$= \frac{(\lambda/\mu)^k \cdot \frac{1}{k!}}{\sum_{k} \left(\frac{\lambda}{\mu}\right)^k \frac{1}{k!}}.$$

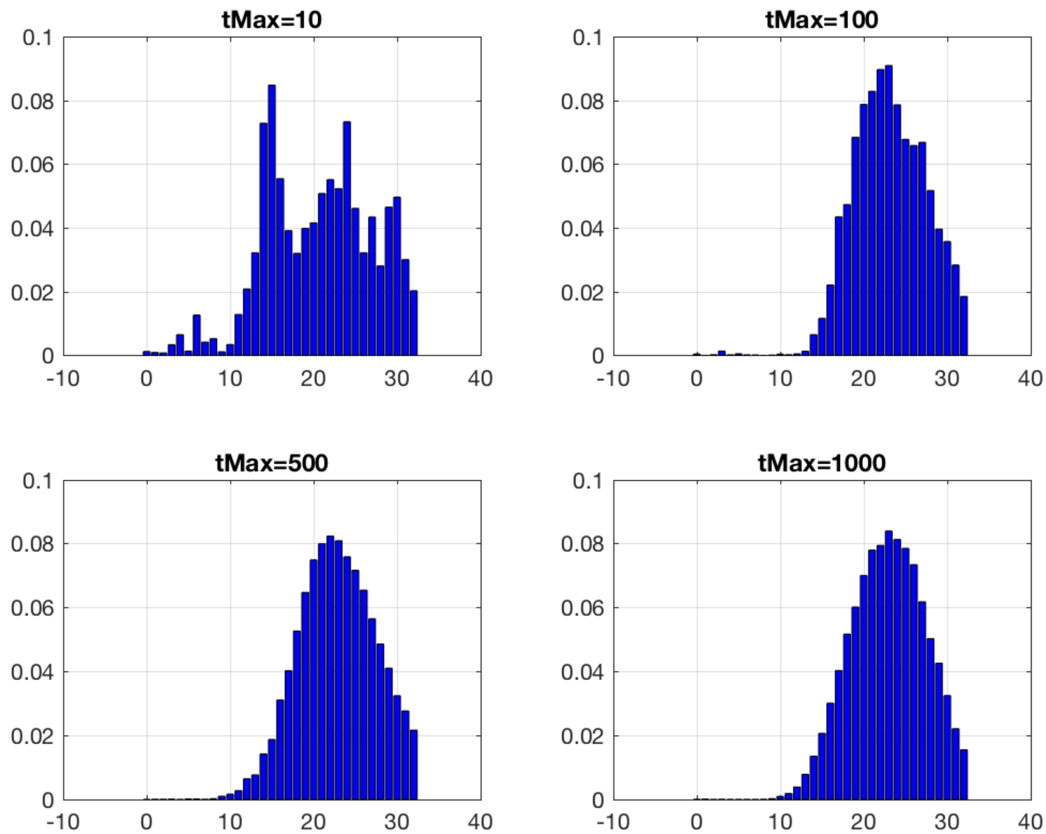
H

Because the CTMC we discussed is irreducible as shown before. The ergotic limit exists. Also, we can formulate the problem to DCMC. The limit will remain unchanged. (5) holds true.

Pk is the value we calculated in G if the process is allowed to run sufficient time.

I

The following graph shows the distribution and its relationship with tMax. Where tMax are of choice [10, 100, 500, 1000]
It is clear that as tMax increases distribution stabilize to a belt-shaped curve.
So we arbitrarily choose 1000 as tMax



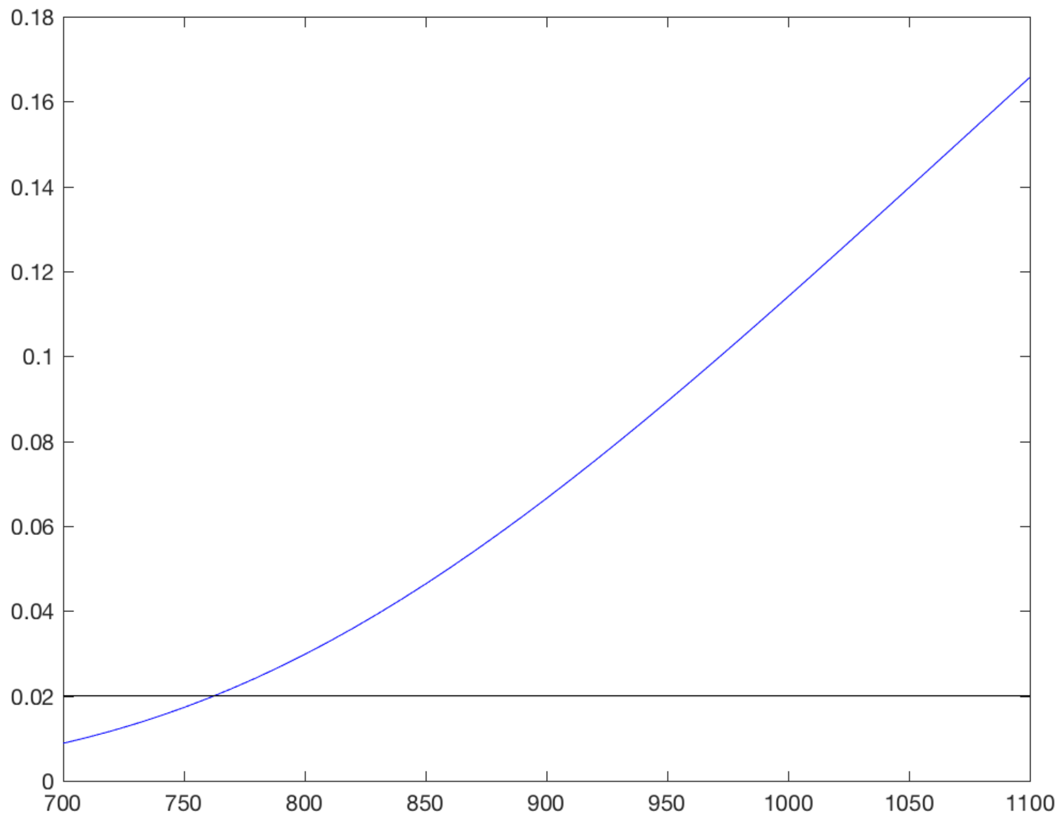
J

The probability of a blocked call is the probability that all channels are occupied:
So, $P=P_k$ as we computed in problem G.

K

Based on the data, we remove two busiest days, also two largest entries. The target date is 10/13, 16:00- 16:30 with 913 calls. This is our target data.
Plus 5% increase: lambda= 958 calls per half hour.

Setting mu and K to be previous number. We simulate the result of block rate changes with incoming calls.



The x axis is the number of coming calls in half hour, y axis is block rate. Horizontal y=0.02 is our tolerance for block rate. It is clear the block rate falls below 0.02 only when calls are below around 760. For all calls larger than that, it is almost certain that block rate will be higher than tolerance, and tolerance increases as an almost linear function of calls. With our target 958 calls, it is certain that it will exceed the tolerance (approximate 0.09 according to the plot). So a new BS is needed.

Code:

```
function final()
    F('/Users/Pagliacci/Desktop/F.png')
    I('/Users/Pagliacci/Desktop/I.png')
    K('/Users/Pagliacci/Desktop/K.png')

end

function F(dir)
    mu=1/(56/60);
    lambda=25;
    TMin=1800;
    TMax=30;
    K=32;
    X0=0;
    [X,T]=HelperF(X0,mu,lambda,K,TMax,dir);
    f=stairs(T,X')
    saveas(f,dir,'png')
end


function [X,T]=HelperF(X0,mu,lambda,K,TMax,dir)
close all; clc
counter=1;
X(counter)=X0;
T(counter)=0;
    while T(counter)<TMax
        x=X(counter);
        if x>=0 && x<K
            taw=exprnd(1/(lambda+mu*x));
            T(counter+1)=T(counter)+taw;
            u=rand;
            if u<((x*mu)/(lambda+x*mu))
                X(counter+1)=x-1;
            else
                X(counter+1)=x+1;
            end
            elseif x==K
                taw=exprnd(1/(x*mu));
                 %appdending result
                X(counter+1)=x-1;
                T(counter+1)=T(counter)+taw;
        else
            break
        end
        counter=counter+1;
    end
end


function I(dir)
    clc; close all;
    X0=0;
    mu=1/(56/60);
    lambda=25;
```

```
    K=32;
    figure(1)
    l=[10, 100, 500,1000]
    for i=1:4;
        subplot(2,2,i)
        HelperI(X0,mu,lambda,K,l(i));
        title(['tMax=',num2str(l(i))])
        grid on
    end
    saveas(figure(1),dir,'png')
end


function [Prob]=HelperI(X0,mu,lambda,K,TMax)
    timePast=zeros(1,K+1);
    x=X0;
    t=0;
    while t<TMax
        if x>=0 && x<K
        taw=exprnd(1/(lambda+mu*x));
        t=t+taw;
        timePast(x+1)=timePast(x+1)+taw;

        u=rand;
        if u<((x*mu)/(lambda+x*mu))
            x=x-1;
        else
            x=x+1;
        end
        elseif x==K
            taw=exprnd(1/(x*mu));
            t=t+taw;
            timePast(x+1)=timePast(x+1)+taw;
            x=x-1;
        else
            break
        end
    end
    Prob=timePast/t;
    bar(0:K,Prob,'b');
end


function K(dir)
    mu=1/(56/60);
    lambda=(700:10:1100);
    K=32;
    figure(2)
    pb=zeros(1,3);
    i=0;
    for L=lambda/30;
        i=i+1;
        pb(i)=((L/mu).^K/factorial(K))/(sum((L/mu).^(0:K)./factorial(0:K)));
    end
    plot(lambda,pb,'b')
    hold on
    plot(lambda,0.02*ones(size(lambda)),'black')
```

```
    saveas(figure(2),dir,'png')
end
```